Automatic Text Summarization

Mohamed Abdel Fattah, and Fuji Ren

***Abstract*—**This work proposes an approach to address automatic text summarization. This approach is a trainable summarizer, which takes into account several features, including sentence position, positive keyword, negative keyword, sentence centrality, sentence resemblance to the title, sentence inclusion of name entity, sentence inclusion of numerical data, sentence relative length, Bushy path of the sentence and aggregated similarity for each sentence to generate summaries. First we investigate the effect of each sentence feature on the summarization task. Then we use all features score function to train genetic algorithm (GA) and mathematical regression (MR) models to obtain a suitable combination of feature weights. The proposed approach performance is measured at several compression rates on a data corpus composed of 100 English religious articles. The results of the proposed approach are promising.

***Keywords*—**Automatic Summarization, Genetic Algorithm, Mathematical Regression, Text Features.

I. Introduction

RECENTLY many experiments have been conducted for text summarization task. Some were about evaluation of summarization using relevance prediction [1], ROUGEeval package [2], SUMMAC, NTCIR, and DUC [3] and voted regression model [4]. Others were about single- and multiple-sentence compression using "parse and trim" approach and a statistical noisy-channel approach [5] and conditional random fields [6]. Some other researches were about multi-document summarization [7], [8] and summarization for specific domains [9] - [11].

In this work, sentences of each document are modeled as vectors of features extracted from the text. The summarization task can be seen as a two-class classification problem, where a sentence is labeled as "correct" if it belongs to the extractive reference summary, or as "incorrect" otherwise. We may give the "correct" class a value '1' and the "incorrect" class a value '0'. In testing mode, each sentence is given a value between '0' and '1'. Therefore, we can extract the appropriate number of sentences according to the compression rate. The trainable summarizer is expected to "learn" the patterns which lead to the summaries, by identifying relevant feature values which are most correlated with the classes "correct" or "incorrect". When a new document is given to the system, the "learned" patterns are used to classify each sentence of that document into either a "correct" or "incorrect" sentence and give it a certain score value between '0' and '1', producing an extractive summary.

Mohamed Abdel Fattah is with Faculty of Engineering, University of Tokushima, 2-1 Minamijosanjima, Tokushima, 770-8506, Japan (e-mail: mohafi@is.tokushima-u.ac.jp).

Fuji Ren is with School of Information Engineering, Beijing University of Posts & Telecommunications, Beijing, 100088, China and Faculty of Engineering, University of Tokushima, 2-1 Minamijosanjima, Tokushima, 770-8506, Japan (ren@is.tokushima-u.ac.jp).

II. Text Features

1) f1 = Sentence Position

We assume that the first sentences of a paragraph are the most important. Therefore, we rank a paragraph sentence according to their position and we consider maximum positions of 5. For instance, the first sentence in a paragraph has a score value of 5/5, the second sentence has a score 4/5, and so on.

2) f2 = Positive keyword in the sentence

Positive keyword is the keyword frequently included in the summary. It can be calculated as follows:

$$Score_{f_2}(s) = \frac{1}{Length(s)} \sum_{i=1}^{n} tf_i * P(s \in S \mid keyword_i) \quad (1)$$

Where, s is a sentence, n is the number of keywords in s, tf_i is the occurring frequency of $keyword_i$ in s. We divide the value by the sentence length to avoid the bias of its length.

$$P(s \in S \mid keyword_i) = \frac{P(keyword_i \mid s \in S) P(s \in S)}{P(keyword_i)}$$

$$P(keyword_i \mid s \in S) = \frac{\#(sentence\ in\ summary\ ,and\ contains\ keyword_i)}{\#(sentence\ in\ summary)}$$

$$P(s \in S) = \frac{\#(sentence\ in\ training\ corpus\ ,and\ also\ in\ summary)}{\#(sentence\ in\ training\ corpus)}$$

$$P(keyword_i) = \frac{\#(sentence\ in\ training\ corpus\ ,and\ contains\ keyword_i)}{\#(sentence\ in\ training\ corpus)}$$

All these values are calculated from the training corpus (manually summarized articles).

3) f3 = Negative keyword in the sentence

In contrast to f2, negative keywords are the keywords that are unlikely to occur in the summary. And it can be calculated as follows:

$$Score_{f_3}(s) = \frac{1}{Length(s)} \sum_{i=1}^{n} tf_i * P(s \notin S \mid keyword_i) \quad (2)$$

4) f4 = Sentence Centrality (similarity with other sentences)

Sentence centrality is the vocabulary overlap between this sentence and other sentences in the document. It is calculated as follows:

$$Score_{f_4}(s) = \left| \frac{Keywords\ in\ s \cap Keywords\ in\ other\ sentences}{Keywords\ in\ s \cup Keywords\ in\ other\ sentences} \right| \quad (3)$$

5) f5 =Sentence Resemblance to the title

Sentence resemblance to the title is the vocabulary overlap between this sentence and the document title. It is calculated as follows:

$$Score_{f_5}(s) = \left| \frac{Keywords\ in\ s \cap Keywords\ in\ title}{Keywords\ in\ s \cup Keywords\ title} \right| \quad (4)$$

6) f6 = sentence inclusion of name entity (proper noun)

Usually the sentence that contains more proper nouns is an important one and it is most probably included in the document summary. The score of f6 is calculated as follows:

$$Score_{f_6}(s) = \frac{\#(proper\ nouns\ in\ s)}{Length(s)} \quad (5)$$

7) f7 = sentence inclusion of numerical data

Usually the sentence that contains numerical data is an important one and it is most probably included in the document summary. The score of f7 is calculated as follows:

$$Score_{f_7}(s) = \frac{\#(numerical\ data\ in\ s)}{Length(s)} \quad (6)$$

8) f8 = sentence relative length

This feature is employed to penalize sentences that are too short, since these sentences are not expected to belong to the summary. We use the relative length of the sentence, which is calculated as follows:

$$Score_{f_8}(s) = \frac{Length(s) * \#(article\ sentences)}{Length(article)} \quad (7)$$

9) f9 = Bushy path of the node (sentence)

The bushiness of a node (sentence) on a map is defined as the number of links connecting it to other nodes (sentences) on the map. Since a highly bushy node is linked to a number of other nodes, it has an overlapping vocabulary with several sentences and is likely to discuss topics covered in many other sentences (Salton, Singhal, Mitra & Buckley, 1997). The Bushy path is calculated as follows:

$$Score_{f_9}(s) = \#(branches\ connected\ to\ the\ node) \quad (8)$$

10) f10 = Summation of similarities for each node (aggregate similarity)

Aggregate similarity measures the importance of a sentence. Instead of counting the number of links connecting a node (sentence) to other nodes (Bushy path), aggregate similarity sums the weights on the links. Aggregate similarity is calculated as follow:

$$Score_{f_{10}}(s) = \sum Node\ branch\ similarities \quad (9)$$

III. The Proposed Automatic Summarization Model

Fig. 1 shows the proposed automatic summarization model. We have two modes of operations:

1- Training mode where features are extracted from 50 manually summarized English documents and used to train GA, MR models.

2- Testing mode where features are extracted from 100 English documents (These documents are different than that were used for training) and go through any of the previously mentioned models to be summarized.

A. Genetic Algorithm Model (GA)

Genetic Algorithms are a way of solving problems by mimicking the same processes Mother Nature uses [12], [13]. Therefore, GA can be used to specify the weight of each text feature.

For a sentence s, a weighted score function, as shown in the following equation is exploited to integrate all the ten feature scores mentioned in section II, where w_i indicates the weight of f_i.

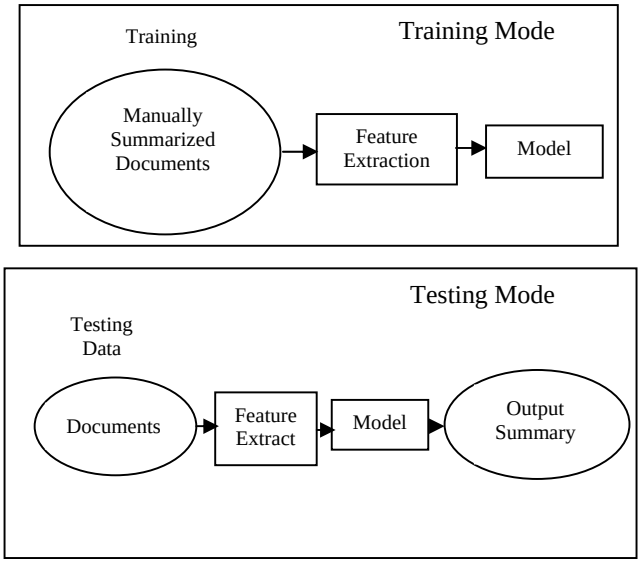

Fig. 1 The proposed automatic summarization model

$$\begin{aligned} Score(s) = {} & w_1.Score_{f_1}(s) + w_2.Score_{f_2}(s) + w_3.Score_{f_3}(s) + \\ & w_4.Score_{f_4}(s) + w_5.Score_{f_5}(s) + w_6.Score_{f_6}(s) + \\ & w_7.Score_{f_7}(s) + w_8.Score_{f_8}(s) + w_9.Score_{f_9}(s) + \\ & w_{10}.Score_{f_{10}}(s) \end{aligned} \quad (10)$$

The genetic algorithm (GA) is exploited to obtain an appropriate set of feature weights using the 50 manually summarized English documents. A chromosome is represented as the combination of all feature weights. 1000 genomes for each generation were produced. Evaluate fitness of each genome, and retain the fittest 10 genomes to mate for new ones in the next generation. In this experiment, 100 generations are evaluated to obtain steady combinations of feature weights. A suitable combination of feature weights is found by applying GA.

For testing, a set of 100 English documents was used. We apply equation 10 after using the defined weights from GA execution. All document sentences are ranked in a descending order according to their scores. A set of highest score sentences are chronologically specified as a document summary based on the compression rate.

B. Mathematical Regression Model (MR)

Mathematical regression is a good model to estimate the text feature weights [14]. In this model a mathematical function can relate output to input. The feature parameters of the 50 manually summarized English documents are used as independent input variables and the corresponding dependent outputs are specified in the training phase. We try to get a relation between inputs and outputs to model the system. Then testing data are introduced to the system model for evaluation of its efficiency. In matrix notation we can represent regression as follow:

$$\begin{bmatrix} Y0 \\ Y1 \\ : \\ : \\ Ym \end{bmatrix} = \begin{bmatrix} X01 & X02 & X03 & ..... & X010 \\ : & : & : & : & : \\ : & : & : & : & :: \\ : & : & : & : & : \\ Xm1 & Xm2 & Xm3 & ..... & Xm10 \end{bmatrix} . \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ : \\ : \\ w_{10} \end{bmatrix} \quad (11)$$

where

[Y] is the output vector.

[X] is the input matrix (feature parameters).

[W] is the linear statistical model of the system (the weights $w_1, w_2,\ldots\ldots w_{10}$ in equation 10).

m is the total number of sentences in the training corpus.

For testing, a set of 100 English documents was used. We apply equation 10 after using the defined weights from [W]. All document sentences are ranked in a descending order according to their scores. A set of highest score sentences are chronologically specified as a document summary based on the compression rate.

IV. Experimental Results

A. The English Data

150 English articles in the domain of religious were collected from the Internet archive. 50 English articles were manually summarized using compression rate of 30%. These manually summarized articles were used to train the previously mentioned three models. The other 100 English articles were used for testing. The average number of sentences per English articles is 32.7.

We use the intrinsic evaluation to judge the quality of a summary based on the coverage between it and the manual summary. We measure the system performance in terms of precision from the following formula:

$$P = \frac{S \cap T}{S} \quad (12)$$

where, P is the precision, T is the manual summary and S is the machine-generated summary.

B. Modified Corpus-Based Approach + Genetic Algorithm (MCBA+GA) of [13]

We are going to exploit the MCBA+GA approach of Yeh et al., for summarization and use it as a baseline approach.

For a sentence s, a weighted score function, as shown in the following equation is exploited to integrate the first 5 feature scores mentioned in section II, where w_i indicates the weight of f_i.

$$Score(s) = w_1.Score_{f_1}(s) + w_2.Score_{f_2}(s) + w_3.Score_{f_3}(s) + w_4.Score_{f_4}(s) + w_5.Score_{f_5}(s) \quad (13)$$

We use the approach as described in section III.A.

For testing, a set of 100 English documents was used. We apply equation 13 after using the defined weights from GA execution. All document sentences are ranked in a descending order according to their scores. A set of highest score sentences are chronologically specified as a document summary based on the compression rate. Table I shows the MCBA+GA approach performance evaluation based on precision using the first 5 features for English documents.

C. The Effect of Each Feature on Summarization Performance

In this section, we investigate the effect of each feature parameter on summarization by using equation 10 with individual score using feature weight equal to 1. For instance, to investigate the first feature (sentence position) on summarization performance, we use the following equation:

$$Score(s) = Score_{f_1}(s) \quad (14)$$

Table II shows the summarization precision associated with each feature for different compression rates for the English documents.

D. The Results of Genetic Algorithm Model (GA)

We have exploited the MCBA+GA approach of Yeh et al., for summarization as described in section IV.B using equation 10 rather than equation 13. Therefore, we have exploited the 10 features for summarization. The system calculates the feature weights using Genetic algorithm.

All document sentences are ranked in a descending order according to their scores. A set of highest score sentences are chronologically specified as a document summary based on the compression rate. Table III shows the GA approach performance evaluation based on precision using the 10 features for the English documents.

E. The Results of Mathematical Regression (MR)

We have exploited the approach in section III.B. For testing, a set of 100 English documents was used. We applied equation 10 after using the defined weights from [W]. All document sentences are ranked in a descending order according to their scores. A set of highest score sentences are chronologically specified as a document summary based on the compression rate. Table IV shows the MR approach performance evaluation based on precision for different compression rates for the English documents.

TABLE I
THE MCBA+GA APPROACH PERFORMANCE EVALUATION BASED ON PRECISION

Compression rate (CR)	10%	20%	30%
Precision (P)	0.4253	0.4265	0.4278

TABLE II
THE SUMMARIZATION PRECISION ASSOCIATED WITH EACH FEATURE FOR DIFFERENT COMPRESSION RATES

Compression rate (CR)	10%	20%	30%
P(f1)	0.3365	0.3487	0.3424
P(f2)	0.3332	0.3318	0.3456
P(f3)	0.2897	0.2975	0.2998
P(f4)	0.3854	0.3993	0.4097
P(f5)	0.3587	0.3548	0.3790
P(f6)	0.3193	0.3285	0.3264
P(f7)	0.2612	0.2698	0.2665
P(f8)	0.2698	0.2652	0.2747
P(f9)	0.4153	0.4176	0.4283
P(f10)	0.3749	0.3682	0.3794

TABLE III
THE GA APPROACH PERFORMANCE EVALUATION BASED ON PRECISION

Compression rate (CR)	10%	20%	30%
Precision (P)	0.4376	0.4435	0.4494

TABLE IV
THE MR APPROACH PERFORMANCE EVALUATION BASED ON PRECISION

Compression rate (CR)	10%	20%	30%
Precision (P)	0.4312	0.4353	0.4382

V. CONCLUSION AND FUTURE WORK

In this paper, we have investigated the use of genetic algorithm (GA), mathematical regression (MR), for automatic text summarization task. We have applied our new approaches on a sample of 100 English religious articles. Our approach results outperform the baseline approach results. Our approaches have been used the feature extraction criteria which gives researchers opportunity to use many varieties of these features based on the used language and the text type. Some text features are language dependent like positive and negative keywords while some other features are language independent.

In the future work, we will investigate the effect of the output summary from this system on information retrieval and cross language information retrieval systems.

ACKNOWLEDGMENT

This research has been partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (B), 19300029, and the Japan Society for the Promotion of Science (JSPS), Grant No. 07077.

REFERENCES

[1] Hobson, S., Dorr, B., Monz, C., & Schwartz, R. (2007). Task-based evaluation of text summarization using Relevance Prediction *Information Processing & Management, 43(6), 1482-1499.*

[2] Sjöbergh, J. (2007). Older versions of the ROUGEeval summarization evaluation system were easier to fool. *Information Processing & Management, 43(6), 1500-1505.*

[3] Over, P., Dang, H., & Harman, D. (2007). DUC in context. *Information Processing & Management, 43(6), 1506-1520.*

[4] Hirao, T., Okumura, M., Yasuda, N., & Isozaki, H. (2007). Supervised automatic evaluation for summarization with voted regression model. *Information Processing & Management, 43(6), 1521-1535.*

[5] Zajic, D., Dorr, B., Lin, J., & Schwartz, R. (2007). Multi-candidate reduction: Sentence compression as a tool for document summarization tasks. *Information Processing & Management, 43(6), 1549-1570.*

[6] Nomoto, T. (2007). Discriminative sentence compression with conditional random fields. *Information Processing & Management, 43(6), 1571-1587.*

[7] Vanderwende, L., Suzuki, H., Brockett, C., & Nenkova, A. (2007). Beyond SumBasic: Task-focused summarization with sentence simplification and lexical expansion. *Information Processing & Management, 43(6), 1606-1618.*

[8] Harabagiu, S., Hickl, A., & Lacatusu, F. (2007). Satisfying information needs with multi-document summaries. *Information Processing & Management, 43(6), 1619-1642.*

[9] Moens, M. (2007). Summarizing court decisions. *Information Processing & Management, 43(6) 1748-1764.*

[10] Reeve, L., Han, H., & Brooks, A. (2007). The use of domain-specific concepts in biomedical text summarization. *Information Processing & Management, 43(6), 1765-1776.*

[11] Ling, X., Jiang, J., He, X., Mei, Q., Zhai, C., & Schatz, B. (2007). Generating gene summaries from biomedical literature: A study of semi-structured summarization. *Information Processing & Management, 43(6), 1777-1791.*

[12] Russell, S. J., & Norvig, P. (1995). Artificial intelligence: a modern approach. *Englewood Cliffs, NJ: Prentice-Hall International Inc.*

[13] Yeh, J., Ke, H., Yang, W., & Meng. I. (2005). Text summarization using a trainable summarizer and latent semantic analysis. *Information Processing & Management, 41(1), 75-95.*

[14] Jann, B. (2005). Making regression tables from stored estimates. *Stata Journal* 5, 288–308.